91602
new 82-10

LEGAL NOTICE

This report was prepared by Ontario Hydro and, while effort has been made to assure the accuracy and applicability of the material contained herein, neither Ontario Hydro nor any person acting on its behalf:

(a) makes any warranty, express or implied with respect to the use of any information, apparatus, method or process disclosed in this report or that such use may not infringe privately owned rights; or

(b) assumes any liabilities with respect to the use of, or for damages resulting from the use of, any information, apparatus, method or process disclosed in this report.

ontario hydro
research division

COMPLIMENTARY COPY
Please Credit Ontario Hydro Research if Reproduced in Whole or in Part

PROGRAMMING SPECIFICATIONS FOR THE OH180 PROGRAMMABLE CONTROLLER SOFTWARE

Report 87-221-K

J.D. Bruce
Engineer
Electronics & Instrumentation
Section
Electrical Research Department

ABSTRACT

This report documents the programming specifications used by the Electrical Research Department in the design and development of software for the OH180 Programmable Controller. The report covers procedures for the design, coding, verification and documentation of the application-independent system software.

job	file	date	report no.
11-160	817.34	October 5, 1987	87-221-K

91601
new 82-10

ontario hydro
research division

To

Mr. M.A. Machacek
Manager of Engineering
Darlington GS, H16G1

PROGRAMMING SPECIFICATIONS FOR THE OH180 PROGRAMMABLE CONTROLLER SOFTWARE

J.D. Bruce

1. INTRODUCTION

This report documents the programming specifications required in the design and development of system software for the OH180 Programmable Controller. It covers the design procedures, coding practices, verification techniques and the methods of documentation that together comprise the OH180 programming specifications. The specifications apply to all OH180 software design and development work carried out by the Electrical Research Department up to the point where the software is released to the Electrical Design Department for functional testing of integrated software. They do not apply to the development of application software (ladder diagrams).

This report forms part of the documentation required for the Software Quality Assurance (SQA) program for the OH180 as called for in the OH180 SQA Manual for Darlington GS/1/. This report was originally circulated in draft form with the number E86-37-K.

2. PRELIMINARY SOFTWARE DESIGN

The general objective of the OH180 system software is to implement the technical requirements specified by the original overall OH180 specification/2/, and related communication system requirements set down in references/3/ through/10/, as well as other changes subsequently requested by the Electrical Design Department.

At the preliminary design stage, the specification shall first be analyzed to determine the aspects that are to be implemented in software, as opposed to hardware.

The system software requirements shall be divided into four major modules, corresponding to the four major hardware components included in the OH180 system:

- Programmable Controller (PC),
- Programming Station,
- Field Monitor (FM),
- Communications Module (CM).

For each of the four, a breakdown of the major software requirements shall be determined, along with the interaction required with the other three modules. In conjunction with the hardware design, the methods required to access any input and/or output hardware shall be determined.

For each of the four major software modules, the detailed technical requirements shall then be established, in accordance with the following subsections, 2.1 through 2.4. Documents shall be prepared defining the detailed technical requirements of each module.

2.1 OH180 Programmable Controller Executive

The fixed, application-independent portion of the software running in every OH180 programmable controller shall be known as the "OH180 PC Executive". Its function shall be to implement the sequencing of operations during start-up, carry out various types of self-testing, keep track of the time, and, every 10 milliseconds, initiate execution of the application-specific "relay logic ladder diagram" code (prepared by the programming station, as described in Section 2.2). The OH180 PC Executive shall also maintain communication with the Communications Module (if used), update the CPU board LED displays, perform various external wiring checks (such as ground fault testing), and interact with the Field Monitor (if connected).

The OH180 PC Executive code shall be written in Intel 8085 assembly language, prepared on a Hewlett-Packard 64000 microprocessor development system. The PC Executive code shall be transferred to the Programming Station by first programming the PC Executive onto an EPROM using the Hewlett-Packard system and then installing the EPROM in an OH180 EPROM module. Using a special function in the Programming Station software, the PC Executive code shall then be read into the PDP-11 and integrated with the Programming Station software, for subsequent downloading to the final OH180 EPROMs along with the application-specific ladder code. For each OH180 PC system the PC Executive shall reside in the first 4 k bytes of address space in the EPROM module.

2.2 OH180 Programming Station Executive System Software

The OH180 Programming Station shall consist of a Digital Equipment PDP-11 computer system, set up to facilitate the generation of "relay logic ladder diagram code" for use in the OH180 Programmable Controllers. The Programming Station software shall

interact with the ladder diagram designer through a keyboard and display, and provide a means of entering and editing ladder diagrams on the display. The software shall provide a means of compiling the ladder diagrams to produce application-specific executable code for the OH180's CPU. It shall also provide a means of generating printouts of the ladder diagrams, tables of references, and timer/counter settings.

The ladder diagrams entered on the Programming Station shall be a symbolic representation of the equivalent relay logic functions required for the various sections of the plant control system being implemented. The inputs, outputs and logic interconnections shall be represented as if they were electromechanical relay contacts and coils, thereby allowing development previously done for traditional relay logic schemes to be used directly or by a straightforward translation.

In order to transfer the generated code to the OH180, the Programming Station shall incorporate EPROM programming software designed to work in conjunction with the OH180 EPROM Programmer. An additional software function shall provide a means of monitoring and testing OH180 Programmable Controllers, in a manner functionally identical to the Field Monitor described in Section 2.3, except for the addition of a capability for downloading a compiled ladder diagram directly to RAM, in suitably equipped OH180 Programmable Controllers for simulation tests.

The Programming Station software shall be implemented in Digital Equipment's FORTRAN IV, to run under the RT-11 operating system. The use of FORTRAN as the coding language is on the basis of it being a software system with all the required functions fully supported by a reputable supplier, and also on the basis of the wide acceptance and knowledge of FORTRAN amongst computer programmers. The Programming Station software shall normally reside on a floppy disk but shall be capable of running from a hard disk.

2.3 Field Monitor Software

The OH180 Field Monitor shall consist of a small PDP-11 computer system enclosed in a video terminal. Unlike the Programming Station, the Field Monitor shall not have a floppy disk drive from which to load its software. In place of the floppy disk, the Field Monitor shall have an EPROM board on which its complete program, including overlays, shall be stored. On power-up and when requested by the software, sections of the stored code shall be transferred from the EPROM into the Field Monitor's RAM memory. Once in the RAM, the software shall operate in the manner outlined below. This operation shall be identical to the corresponding monitoring and testing function in the Programming Station.

The Field Monitor software shall initially provide the user with two options:

- display all faults that exist on the OH180 PC to which it is attached,
- display a dynamically updated ladder diagram, as currently operating in the OH180 PC.

In either case, the Field Monitor software shall send a series of commands to the connected OH180 Programmable Controller and receive, in reply, data from the OH180 PC's memory which shall be decoded to establish the requested display.

The Field Monitor's fault data display shall indicate the results of the self-tests and I/O testing continuously carried out by the OH180 PC Executive. The information shall be in the form of clear and unambiguous messages that the user can easily interpret and use while checking out the operation of an OH180 PC.

To recreate the ladder diagram, the Field Monitor software shall read and disassemble all of the information in the OH180 PC's EPROM memory. Based on this information it shall redraw the ladder diagram on the video display. The ladder diagram elements, labels, text entries and descriptive comments originally entered by the ladder diagram designer shall be recovered from the OH180 PC and displayed. The reconstructed display shall be virtually identical to the original ladder diagram (but shall be rebuilt without having to access the ladder designer's floppy disk files or any information other than what is stored in the OH180 PC's EPROM module).

In addition, the Field Monitor software shall superimpose the status of all the displayed elements. Coils that are energized and contacts that are in the active state shall be highlighted on the screen. The present time and count values of the timers and counters shall be dynamically updated. The user shall be able to scroll up and down the display, or request that a particular portion of the ladder diagram be displayed. In addition the user shall be able to temporarily override the status of contacts and coils to force them into a specified state. The user shall also be able to temporarily redefine the settings for timers and counters. All such forcing functions shall automatically revert to their original values whenever the Field Monitor is disconnected.

The Field Monitor software shall be written primarily in FORTRAN, with certain routines written in PDP-11 MACRO assembly language where necessary to implement special functions requiring higher execution speed than is possible in FORTRAN. The "bootstrap" routine used to load the program in from EPROM on power-up may also be written in assembly language, if impractical in FORTRAN. The Field Monitor software shall be written to run under a diskless version of the RT-11 operating system called SIMRT. Extensions to SIMRT to support virtual overlays and virtual arrays shall be implemented in PDP-11 MACRO assembly language.

2.4 Communications Module Software

The OH180 Communications Module software shall be designed to exchange data with a host OH180 PC and to transmit and receive data over a dual-ring fiber-optic communications system to and from other OH180 Communications Modules, each having its own host OH180 PC.

Through interaction with the host OH180 PC, the Communications Module software shall generate a table containing the origin of any external communications data referenced in the ladder diagram. Using this, the Communications Module shall keep records of all received data relevant to its host and periodically provide this information to the OH180 PC. All valid data received from other units shall be retransmitted to the next sequential "down-stream" Communications Module on the ring. In addition, the Communications Module shall add the data from its host OH180 PC to the message stream. If the loop is intact, the locally generated message will eventually work its way around the loop and be received as an incoming message. When this occurs, the message shall be checked and removed from the data stream.

The two loops in the dual ring shall normally link the same Communications Modules in opposite directions in order to provide a redundant source for data. The second loop shall operate completely independently of the first. As the Communications Module will normally receive data from the same source on both loops, the software shall be able to recognize the most up-to-date data, by means of a retransmission counter included in each message.

The Communications Module software shall also provide error checking on all messages and self-testing of its own hardware. It shall check for missing incoming data, in order to detect if other modules on the loop have failed. The results of these tests shall be included in the messages transferred to the host OH180 PC.

The Communications Module software shall be written in Intel 8085 assembly language on a Hewlett-Packard 64000 microprocessor development system. The software shall be application-independent and shall be permanently installed in an EPROM in the Communications Module.

3. DETAILED SOFTWARE DESIGN

3.1 General Considerations

In designing the OH180 software, the general principles of structured programming shall be followed. Starting from the technical requirements, each of the four major software modules shall be subdivided into separate clearly defined submodules. The task of each submodule shall be identified and the required data defined. A determination shall be made as to what data is for local use and what data is to be acquired from and/or distributed to other submodules.

Structures for storing the data for each submodule shall be designed such that data common to more than one submodule uses the same format (or a compatible format) in all submodules.

In implementing the principles of structured programming, the following specific guidelines shall be adhered to, wherever practical:

- Top down programming (modules, submodules, sub-submodules etc, each with well defined functions)
- Single entry points and returns (ie no jumping from one submodule to another, except by its standard entry/exit procedure)
- Clearly identified boundaries between submodules (for the case where two or more submodules are located in the same source file)
- Global or common variables clearly identified as to function, and never used for more than one purpose
- Embedded constants avoided where possible

Although undesirable, it may be necessary in some cases to depart somewhat from the principles of structured programming in order to handle constraints of execution time or programming language. For example, since RT-11 FORTRAN does not support "ELSE" structures, the use of "GO TO" may be unavoidable in certain cases. It may also be necessary for certain submodules to be physically arranged as in-line code, rather than subroutine structures.

In all such departures, however, the code shall be written and commented in such a way that the structure is clear and obvious.

3.2 FORTRAN Coding Practices

All routines written for use on the PDP-11 shall be written in FORTRAN IV under the RT-11 operating system except for routines for which timing is critical or for which existing assembly language software meeting the assembly language coding requirements outlined below is available.

Wherever practical the variable names used in the FORTRAN code shall provide an indication of the function served by that variable. The usage of all variables exchanged between routines in COMMON blocks or through subroutine and function parameters shall be clearly defined near the start of all routines in which they are referenced.

Any local variables for which the usage is not self-evident shall be defined near the start of the routine in which they are used.

Constants having a defined purpose shall be assigned names and defined in a DATA statement either in the routine, if the con-

stant is referenced in only one routine, or in the BLOCK DATA submodule when referenced in more than one routine. The constant shall always be referred to by name in the code. This assures that any change to the value of the constant is applied to all places where the constant is used.

All irregularities in the type or size of arrays, or the structure of common blocks shall be described in comments located near the array or common block definition.

Where feasible, subroutines shall have a single clearly defined function. Exceptions to this practice shall be made only when the memory space is limited. (FORTRAN IV under RT-11 creates a table of "vectors" for all multidimensional arrays in each routine where the array is defined even if the array is in a common block and even if there are other arrays having similar vector tables within the same addressing range. Where memory usage is critical, these frequently sizable vector tables may be avoided by merging subroutines with the calling routine.)

All routines shall be heavily commented so that the function of the routine and the methods of implementing that function are made clear. The comments shall be of such a nature that they describe how the code works in the context of the routine, or what its role is in the overall operation of the OH180 system, and shall not simply explain in text what a specific FORTRAN statement does (except as noted below).

In cases where the FORTRAN code is being used in a non-standard way (e.g. the application of assigned GOTO statements to provide a method of accessing memory and "I/O page" addresses which are inaccessible using normal FORTRAN coding techniques) the usage shall be clearly defined step by step, and shall include an explanation of why the non-standard technique was used.

As a rough guide, each source file should have at least as many characters in comments as characters in the FORTRAN code itself. Sections of code which are complex shall have more comment than code. Simple, self-evident coding or redundant coding structures can be commented to a lesser extent.

All FORMAT statements shall be located near the I/O statement that references them so that the function and techniques used in the I/O operation are evident when reading through the program listings.

The compiling and linking procedures for the FORTRAN-based modules shall be documented in the form of command files. (The command file, in addition to documenting the required procedures, will also serve to simplify and standardize the compiling/linking/loading process.)

A record of all changes made, known problems, the nature of the corrections and the dates on which they are made, shall be maintained. Source comparison ("SRCOM") listings showing the specific changes between various software releases shall be kept on file.

3.3 8085 Assembly Language Coding Practices

All software modules intended for use on the 8085 microprocessor shall be written in Intel 8085 assembly language. The following guidelines shall be used for all assembly language programs. All program and variable names shall be assigned so that they are easily related to their functions. In addition, each name shall preferably be 6 characters long (or less), and if longer than 6 characters shall have the first 6 characters unique.

Program submodules shall be coded using simple logic constructions as much as possible. This requirement does not apply to time-critical coding where non-standard or optimized code is necessary. However, all such uses of non-standard code shall be well commented so as to make the structure clear and obvious.

Comments shall be freely used to explain the operation of the program code. Comments are considered to be an essential part of the source code.

Each assembly language module shall begin with the following sections:

- identifier
- editing history
- equates
- data definitions

Each assembly language module shall be identified by a unique identifier which shall consist of the module name, date, version number and revision number.

An editing history shall be maintained in each assembly language module. The editing history shall contain the date and description of changes made to the module.

All constants, such as memory addresses, table sizes and peripheral commands shall be defined using equate ('EQU') statements. The equate statements shall be grouped together in a section located before the beginning of the program code. Each equate statement (consisting of the name, the 'EQU' statement and the value of the constant) shall be followed by a comment describing the constant.

All data items shall be declared in a program section located before the program code. The data declarations shall be separated into non-volatile memory (EPROM) definitions and read/write memory (RAM) definitions. Each data declaration (consisting of the name and the declared size of the data item) shall be followed by a comment describing the data.

The program code section shall contain all executable 8085 assembly language code. The following program code sections shall be clearly labelled (if used):

- interrupt service vectors
- interrupt service routines
- main (or background) program
- subroutines

As required, each of the above sections shall be further subdivided into a series of clearly labelled submodules, each with a single well-defined function.

4. SOFTWARE VERIFICATION

Verification of the OH130 software shall be done at several levels, as outlined in the following Sections 4.1 through 4.3. These steps shall be carried out for the initial version of the software as well as all subsequent additions and revisions.

4.1 Technical Reviews

The technical requirements, as outlined in Section 2 and clarified in the resulting "technical requirements" documents shall be reviewed, at the peer level and at the supervisor level, for completeness and correctness in implementing the original overall specification.

Following this, the resulting software designs, structures and, if applicable, flow diagrams, shall be reviewed for consistency with the technical requirements and for adherence to the general principles of Section 3.

4.2 Mechanical Checks of Coding

After each item of source code is written, it shall be entered, edited, compiled, linked and loaded. By using the standard error checking features of the applicable assembler and/or compiler, linker and loader, the software shall be checked for mechanical coding errors and inconsistencies. The software shall be re-edited and the process repeated as required, until all such mechanical errors are eliminated.

4.3 Integration and Functional Tests

At this point the software shall be tested using the steps prescribed in the OH180 Software Development and Integration Test Procedure/11/. Briefly, this involves first testing the individual submodules for correct operation, using suitably constructed test data. In the case of software running in microprocessor chips, use can be made of an in-circuit emulator, where applicable. Following this, the submodules shall be integrated, and the integrated modules shall be checked out, at first individually, and later in conjunction with the other

modules. The final step in the software testing shall be an integrated test of all the modules together in the context of several test ladder diagrams and applicable hardware configurations. The integrated test should be devised so that all software paths are exercised.

During development, the software submodules may fail at some point in the foregoing process. When this happens the software shall be revised as required, and the foregoing steps repeated (starting with Section 4.1).

5. SOFTWARE RELEASE AND CONFIGURATION MANAGEMENT

5.1 General

When a particular software module or group of modules has reached the point where it can pass the development tests of the foregoing section, a software release shall be prepared containing the the most recent versions of all the OH180 system software. The final integrated tests referred to in Section 4.3 shall then be carried out using the software in the exact form as it is to be released.

If this final test passes, the software shall be released to Electrical Design Department for functional testing of the integrated software, using actual application ladder diagrams. The release shall be accompanied with a set of release notes describing the new software and/or changes made to the software since the previous release. The requirements for the release notes are given in Section 5.3.

5.2 Version Control

Each version of each source file of software shall be assigned a unique identifier to distinguish it from other versions of the same source file. No two different versions of the same software shall have the same version identifier.

5.3 Release Notes

In general, release notes shall:

a) identify which aspects of the software are being released, including version identifiers and media labels;

b) document (by "SRCOM" listings, by directory listings, by complete transfers or by other appropriate means) the full and detailed extent of any and all changes to the source code since the previous release;

c) describe functional changes, additional features and bug fixes since previous releases, along with references to other documentation, if applicable;

d) identify compatibilities of hardware and software configurations relative to previous revisions;

e) identify special operating restrictions or instructions (if applicable); and

f) identify known bugs, limitations or problems (if any).

5.4 Archiving

Complete copies of all releases shall be stored in two different places; namely, at the Research Division and at Head Office. In cases where the full source code is not part of the release, it is not necessary to store copies of the source code in two different places, provided:

a) the source code changes from previously released sources are fully documented in the release notes; and

b) the changes from the previously released sources are minor (ie would require not more than a few minutes of editing time to re-create from the previous archived version).

6. FUNCTIONAL TESTING OF INTEGRATED SOFTWARE

As mentioned in Section 5.1, the release shall be issued to an independent group (ie Electrical Design Department) for functional testing of the integrated software. Although the details of this testing are outside the scope of this report, they involve functional testing in the context of the complete OH180 system and shall be designed to verify that the software works correctly in realistic simulations of planned OH180 installations, using actual application ladder diagrams. This testing shall be continued until the independent testing group is satisfied that the system is completely functional or until problems are identified.

Notification of all detected problems and/or suggestions for changes shall be documented and directed back to the software development group (Electrical Research) who shall implement the required changes and then re-issue the software as per Section 5. No changes shall be implemented to the software by the independent testing group.

7. ERRORS, MODIFICATIONS AND/OR ADDITIONS

All error reports and requests for modifications and/or additions to the function of the code sections shall be documented and sub-

mitted to the OH180 system software project coordinator (Electrical Research) who shall decide how the changes are to be implemented. The coordinator shall convey the error report or request to the programmer(s) responsible for the module(s) affected.

All modifications and additions shall be implemented and tested to the same standards and using the same procedures as with the original code.

8. ONGOING SUPPORT CAPABILITY

Equipment shall be kept available at the Research Division to permit the complete operation of the OH180 system software, including an OH180 PC, the Programming Station, Field Monitor and Communications Modules, in order to investigate reported software problems and to demonstrate and test modified versions of the software.

9. DOCUMENTATION

In addition to the various documents cited in this report, manuals shall be prepared describing the OH180 system software in detail and, where applicable, the related hardware aspects. The manuals shall include (or reference) the particular version of software upon which the manual is based. The manuals shall be updated to correspond with updated versions of the software, as required.

Submitted:

J.D. Bruce
Engineer
Electronics & Instrumentation Section
Electrical Research Department

Approved:

A.F. Baljet
Manager
Electrical Research Dept

JDAG:vc

Reviewed:

J.D.A. Griffin
Unit Head - Electronics
Electronics & Instrumentation Section
Electrical Research Department

REFERENCES

1. Software Quality Assurance (SQA) Manual for Ontario Hydro's OH180 Programmable Controller: Darlington GS; Electrical Design Department, November 27, 1986.

2. Specification STD5-X-H-60800-0044 for OH180 Programmable Controller.

3. Memo from R.D. Brown to I.B. Berezowsky, "OH180 Communication System Alternatives", January 20, 1984.

4. Memo from R.J. Ross to I.B. Berezowsky, "OH180 Communications Development", January 26, 1984.

5. Memo from R.D. Brown to I.B. Berezowsky, "OH180 Communication System Alternatives", February 15, 1984.

6. Notes of Meeting - OH180 Communication System - Technical Overview, April 4, 1984.

7. Memo from I.B. Berezowsky to M.A. Machacek, "Darlington GS A OH180 PC Communication Board Recommendation", April 17, 1984.

8. Minutes of Meeting - OH180 Communication System, May 3, 1984.

9. Minutes of Meeting - OH180 Communication System, June 6, 1984.

10. "OH180 Communications Module Technical Manual", H.E. Whitmell, Ontario Hydro Research Report 85-255-H, July 18, 1986.

11. J.D.Bruce, "OH180 Programmable Controller Software: Development and Integration Test Procedures", Electrical Research Department Report No. 87-222-K.